AMENDMENTS TO THE SPECIFICATION

Please replace paragraph [0005] with the following amended paragraph:

[0005] Unfortunately, there is always some risk of unauthorized users obtaining an authorized user's credentials and using the credentials to impersonate the authorized user. Since an authorized user's credentials essentially allow full access to all of authorized users resources on a particular system (e.g., files, electronic messages, personal and financial data, etc.), any ~~comprise~~ compromise in credentials can provide an unauthorized user with the ability to copy and destroy the authorized user's resources. In particular, passwords are vulnerable to guessing attacks, for example, from programs that sequentially submit each word in a dictionary as a password (commonly referred to as "dictionary attacks").

Please replace paragraph [0014] with the following amended paragraph:

[0014] In other embodiments, a server sends a first request that includes at least the authentication mechanisms deployed at the server computing system. The client receives the first request and sends a first response that includes at least the authentication mechanisms deployed at the client computing system. The client and server identify a tunnel key that can be used to encrypt content transferred between the client computing system and server computing system.

Please replace paragraph [0015] with the following amended paragraph:

[0015] The ~~[[sever]]~~ server sends a second request that includes encrypted authentication content (encrypted with the tunnel key) indicating a mutually deployed authentication mechanism. The client receives the second request and decrypts the encrypted authentication content with the tunnel key to reveal unencrypted authentication content. The unencrypted authentication content indicating the mutually deployed authentication mechanism. The client sends a second response including encrypted response data that is the response to the unencrypted authentication content. The encrypted response data contains information for authenticating with the server according to the mutually deployed authentication mechanism. The server receives the second response including the encrypted response data that contains information for authenticating with the server according to the mutually deployed authentication mechanism.

Please replace paragraph [0027] with the following amended paragraph:

[0027] Figure 1 illustrates example computer architecture 100 that facilitates more efficient and secure authentication of a computing system in accordance with the present invention. As depicted in computer architecture 100, client computing system 101 includes key pair 103. Key pair 103 includes public key 104 and corresponding private key 106, for example, a Diffie-Hellman key pair. Server computing system 111 includes credential provisioning module 112 and key pair 113. Credential provisioning module 112 can be configured to receive a first type of credential, such as, for example, a limited-use credential, and, based on the first type of credential, provision a second type of credential, such as, for example, a more permanent credential. Similar, to key pair 103, key pair 113 includes public key 114 and corresponding private key 116, for example, a Diffie-Hellman key pair.

Please replace paragraph [0029] with the following amended paragraph:

[0029] The use of a limited-use credential can be limited in any number of ways. For example, limited-use credential can be valid for a specified number of uses, for a specified period of time, or until the occurrence of a specified event. A limited-use credential can be limited to any number of valid uses (e.g., three uses), based on applicable security policies. Limited-use credentials that are valid for authenticating only once may be referred to as "single-use credentials". After the specified numbers of uses, the limited-use credential is no longer accepted as a valid credential.

Please replace paragraph [0034] with the following amended paragraph:

[0034] Similarly, a client computing system can encrypt a trust anchor using an encryption key derived from a shared secret and the Diffie-Hellman session key and send the encrypted trust anchor to a server computing system. Accordingly, when the server computing system receives the encrypted trust anchor, the server computing system can decrypt and validate the trust anchor. A trust anchor can include authentication related data, such as, for example, a certificate, (e.g., an X.509 certificate), a security token (e.g., a WS-Security token), a hash (e.g., SHA-1) and Uniform Resource Identifier ("URI") (e.g., a Uniform resource Locator ("URL")) of a certificate, or a hash and URI of a security token.

Please replace paragraph [0044] with the following amended paragraph:

[0044] Figure 3 illustrates a message exchange 300 for negotiation authentication mechanisms. ~~[[In]]~~ It should be understood that message exchange 300 can occur before or after the exchange of other messages during authentication. For example, a client computing system and server computing system can exchange one or more Extensible Authentication Protocol ("EAP") request/response pairs that preliminarily identify the client computing system and server computing system to one another.

Please replace paragraph [0045] with the following amended paragraph:

[0045] The requests and responses depicted in message exchange 300 can be messages of an authentication protocol. Each message can ~~includes~~ include the version number of the authentication protocol (e.g., representing supported payload types), a message body, and a Hashed Message Authentication Code ("HMAC") of a portion of the message body. An HMAC can be generated using any cryptographic hash function, such as, for example, MD5, SHA-1, etc. The messages of the authentication protocol can be embedded within EAP messages.

Please replace paragraph [0046] with the following amended paragraph:

[0046] Server side 360 can send server request 301 to client side ~~[[305]]~~ 350. Server request 301 includes previous packet ID 302, nonce 303, and authentication methods 304. Previous packet ID 302 can indicate the packet ID corresponding to the last packet that was exchanged between client side 350 and server side 360 (e.g., the packet ID of packet in a previous request/response exchange). Nonce 303 can be random data generated at server side 360. Authentication methods ~~[[305]]~~ 304 can include the proposed authentication mechanisms supported at server side 360. A server side can support any number of different authentication mechanisms (e.g., challenges and responses as previously described, MS-CHAP v2, Authentication with MD5, Authentication with Generic Token Card, Authentication with Kerberos, Authentication with X.509, and Authentication with WS-Security).

Please replace paragraph [0047] with the following amended paragraph:

[0047] In response to server request 301, client side 350 can send client response 306 to server side ~~[[306]]~~ 360. Client response 306 can include previous packet ID 307, nonce 308, security association(s) 309, public key(s) 311, and authentication methods 312. Previous packet ID 307 can indicate the packet ID corresponding to server request 301. Nonce 308 can be random data generated at ~~client~~ server side 360. Security Associations(s) 309 can include proposed security associations that are supported at client side 350. Table 1 indicates some of the security associations that can be supported.

Please replace paragraph [0057] with the following amended paragraph:

[0057] ~~[[In]]~~ It should be understood that other types of encrypted authentication content (instead of negotiation encrypted content 318 or re-authentication encrypted content 328) can alternately be included in server request 313. For example, when bootstrapping a client using an existing username and password, server request 313 may have encrypted content including an authentication signature, an identity certificate, and an authentication method.

Please replace paragraph [0061] with the following amended paragraph:

[0061] Client side 350 can generate an appropriate response to challenge 319. For example, an appropriate response can be the HMAC of the challenge ~~[[119]]~~ 319 using a shared secret. An appropriate response can be configured according to the following formula:

$Response_c = HMAC_{ss}$ [Challenge]

Please replace paragraph [0067] with the following amended paragraph:

[0067] With reference to Figure 4, an example system for implementing the invention includes a general-purpose computing device in the form of computer system 420, including a processing unit 421, a system memory 422, and a system bus 423 that couples various system components including the system memory 422 to the processing unit 421. Processing unit 421 can execute computer-executable instructions designed to implement features of computer system 420, including features of the present invention. The system bus 423 may be any of several types of bus structures including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of bus architectures. The system memory includes read only memory

("ROM") 424 and random access memory ("RAM") 425. A basic input/output system ("BIOS") 426, containing the basic routines that help transfer information between elements within computer system 420, such as during start-up, may be stored in ROM 424. ~~[[.]]~~